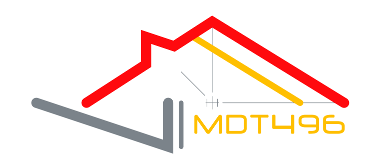


MDT496 ESTUDIOS
DEPARTAMENTO DE OBRAS LITERARIAS
COLECCIÓN: DOCUMENTOS DE NORMALIZACIÓN
ISO - Unicode - SI - IEC - ITU - AISC - ACI

Unicode

Una breve historia sobre codificación de caracteres

Marco Mendieta Parihuancollo

Primera Edición

annlimma.atwebpages.com
youtube.com/mdt496
mdt496@gmail.com
Oruro, Bolivia

Unicode - Una breve historia sobre codificación de caracteres
Marco Mendieta Parihuancollo
Primera Edición (mdt496-ID: 036MP.AABD.009)



Dedicatoria

Esfuerzo sereno, triste pero siempre altivo, por encontrar en la energía misma de la vida el camino para aceptar la muerte.

Amado padre, hoy ausente, agradezco las muchas horas que estuviste a mi lado otorgando tu apoyo y protección. Al final, un silencio implacable, solo recuerdos que arañan el alma.

Este trabajo, producto de un gran esfuerzo, se lo dedico a mi amado padre **Marcos E. Mendieta C.**(†).

Contenido

Lista de tablas

Lista de figuras

Glosario

A

American Standard Code for Information Interchange (ASCII) Forma de ordenar los caracteres que se pueden escribir utilizando un formato de 7 bits., pág. 5.

U

Unicode Unicode es un estandard creado en 1991 que pretende uncluir todos los caracteres de uso común. Un repositorio de más de 100 mil caracteres; Planillas de códigos para referencia visual (Code Charts); Metodologías de codificación; Codificaciones estándares; etc. The Unicode Standard / the Unicode Consortium; edited by the Unicode Consortium. Unicode and the Unicode Logo are registered trademarks of Unicode, Inc., in the United States and other countries., pág. 19.

Unicode Transformation Format 8-Bit (UTF-8) UTF-8 can be used to transmit text data through communications systems that assume that individual octets in the range of x00 to x7F have a definition according to ISO/IEC 4873, including a C0 set of control functions according to the 8-bit structure of ISO/IEC 2022. UTF-8 also avoids the use of octet values in this range that have special significance during the parsing of file name character strings in widely used file-handling systems., pág. 96.

Universal Multiple-Octet Coded Character Set (UCS) This 32-bit character encoding standard is for all practical purposes identical to Unicode standard. The layout of the Basic Multilinula Plane (plane 0 or BMP) is described in detail. This architecture is specifies by [1]. Complementing [2], which describes plane 0 (BMP) of the UCS, the present standard details the layout of the supplenentary planes;, pág. 96.

Introducción

Este libro describe el concepto de carácter, sistemas de codificación de caracteres y el estándar Unicode. El libro está dividido en cuatro capítulos.

En el capítulo 1 presentamos a los lectores las bases de la codificación de caracteres enfocado al desarrollo de software. El capítulo ilustra el concepto de carácter y sus atributos como *glifo* y *punto de código*; Justificamos su importancia y los efectos que produce el ignorarlo; Por último, presenta una reseña del proceso de intercambio de información, que sirve de prefacio al tema sistemas de codificación. Tras leer este capítulo, debería tener el conocimiento básico, que necesitará para entender el resto de este libro.

En el capítulo 2 se describe algunos sistemas de codificación. De cada una veremos atributos como, el nombre oficial (el que es designado por un organismo internacional) el canónico, es decir, por el que es más conocido y sus seudónimos (aliases). La sección de comparativa no se trata de listar ventajas o desventajas de una tabla X, nos enfocaremos en mostrar coincidencia en *puntos de código* de la tabla X con la Y. Por ejemplo, Latin-1 y Windows-1252 coinciden en 96 de 128 caracteres que representa el 75 % cuyas posiciones 160 al 255.

El capítulo 3 trata del estándar Unicode. En su cuerpo nos explican un poco más sobre glifos o el uso de términos *code point* y *UTF-16 code unit*; Mostrar algunas de las plantillas que podrían ser útiles, por ejemplo: Los números 0-9 (U+0030 al U+0039) y las letras A-Z (U+0041 al U+005A), a-z (U+0061 al U+007A) pertenecen a la familia BASIC LATIN (U0000); La letra minúscula u con diéresis ü (U+00FC) está en el bloque LATIN-1 SUPPLEMENT (U0080); Algunas letras griegas como α (U+03B1), β (U+03B2), γ (U+03B3) están en la plantilla GREEK AND COPTIC (U0370). *Note el argumento entre paréntesis después del nombre de un grupo, esto no es casual, representa el identificador del primer carácter con el que inicia el bloque por lo tanto, es el identificador del bloque. Esto facilita la búsqueda porque le da el nombre al archivo en PDF, en este caso* ***U0370.pdf***. Algunos caracteres cirílicos como Ж (U+0416) o л (U+043B) están en el bloque CYRILLIC (U0400).

El capítulo 4 veremos y aplicaremos todo lo aprendido en situaciones reales, por ejemplo,

```
\documentclass{article}
\usepackage[papersize={55mm,16mm}, margin=1mm]{geometry}
\usepackage{lmodern,xcolor}
\usepackage[T2A, T1]{fontenc}
\begin{document}
    \centering
    \char76\char111\
    \char113\char117\char101\
    \char102\char117\char101\
    \char118\char225\char108\char105\char100\char111\
    \char97\char121\char101\char114\
    \char115\char101\char114\char225\
    \char109\char97\char241\char97\char110\char97\
    \char111\char98\char115\char111\char108\char101\char116\char111\char46
```

```
    \color{purple}\fontencoding{T2A}\fontfamily{cmtt}\selectfont
    \char198\char229\char235\char224\char229\char236\
    \char226\char224\char236\
    \char243\char228\char224\char247\char232\char33
\end{document}
```

Lo que fue válido ayer será mañana obsoleto.

Желаем вам удачи!

el cual es un archivo LaTeX que utiliza conocimientos de codificación de caracteres, ya que es independiente de él, y conocimientos sobre codificación de fuentes, dado que los caracteres fueron reemplazados por secuencias del comando `\char` el cual invoca un carácter de una colección cuya posición coincida con la que recibe en su argumento. Esto puede verse como una herramienta de ofuscación, privacidad, seguridad o integridad de la información durante la transferencia de archivos de texto plano. En este caso nos sirve para despedir esta breve introducción con la frase en ruso cuya traducción es: **Le deseamos buena suerte!**

Capítulo 1

Cosas que debe saber

Abordaremos partiendo de Java, porque este libro es un complemento al libro *Programación en Java* del mismo autor, al igual que el capítulo *Expresiones regulares* del mencionado libro, pero a diferencia de este, el tema Unicode es más extenso y por eso se decidió separarlo, además se me hace más fácil explicar por este modo. Si usted no programa en Java, no se preocupe, la sintaxis del lenguaje son tangentes de la historia que pretendo contarle. Pues bien, empecemos.

1.1 Lo básico

En Java, los caracteres están a cargo de la clase `Character` (`java.lang`) cuyo primitivo es `char`, el acepta asignaciones de *literal de carácter* y *literal numérico entero*. Un literal de carácter se expresa como un carácter o una secuencia de escape, encerrado entre comillas simples.

```
char w = 'Å';       // Literal de carácter único
char x = '\u00C5';  // Literal de carácter, escape Unicode (UTF-16)
char y = '\305';    // Literal de carácter, escape Octal (\u0000 a \u00ff)
char z = 197;       // Literal numérico entero
```

Como pudo ver, en Java el identificador de un carácter es el mismo pero expresado en diferentes sistemas numéricos, octal, decimal y hexadecimal. Además, nótese que dos de ellos son representadas mediante secuencias de escape (`'\305'` y `'\u00C5'`); No obstante tambien pueden ser asignados como literales numéricos cuya escritura es `0305` (octal) y `0x00C5` (hexadecimal). La diferencia solo tiene lugar en el octal, de ingresarse como secuencia de escape soporta el rango entre 0 a 255.

Un carácter Unicode a su vez cuenta con varios atributos, por mencionar algunos: *glifo*, *nombre*, *identificador*, *familia* y *categoría*.

El *glifo* es la representación gráfica final (en pantalla o impresa en un papel). El estándar de Unicode específica una figura de referencia del cómo debe dibujar el carácter (si es que es dibujable).

```
char x = '\u221E';
System.out.printf("glifo: %s\n", x);
System.out.printf("name: %s\n", Character.getName(x));
```

```
glifo: ∞
name: INFINITY
```

Es de referencia ya que la forma final queda a disposición de una implementación en particular, por ejemplo: La tipografía, el cual define en base la fundición (Computer Modern,

Sans Serif, Times New Roman), tamaño (13, 14, 15 pt) o estilo (negrita, itálica); No satisfecho con la tipografía, puede usted mismo definir su propia implementación, esto lo veremos detalladamente en el capítulo 4.

El *nombre* es el común por el cual es más conocido o la que mejor la describen sus predecesoras (ASCII, Latin-1, Windows-1252, ISO 6429, etc.).

El *identificador*, conocido como *punto de código*, número que lo identifica a nivel del estándar en específico. Por ejemplo, el carácter INFINITY en la tabla IBM 437 está en la posición 236, en la tabla ASCII no existe (porque esta tabla solo tiene 128 caracteres) y para el estándar Unicode tiene la posición U+221E.

```
char x = '\u221E';
System.out.printf("codePoint: %d\n", (int) x);
System.out.printf("codePoint: U+%04X\n", (int) x);
```

```
codePoint: 8734
codePoint: U+221E
```

En este punto es necesario aclarar a los blogueros, YouTuber's y demás personas que desconocen estos temas; nombrar "ASCII extendido" a una tabla que tiene 256 caracteres es un **error**. La tabla con nombre "ASCII extendido" **¡no existe!** Lo que *sí* existe son tablas, que, basándose en la ASCII original, de 128 caracteres (posiciones 0 al 127, 7 bits con 1 bit libre), definen sus propios 128 caracteres (posiciones 128 al 255, implementación del bit libre que dejo ASCII) resultando los 256 caracteres antes mencionados. Siendo un poco más técnicos, 11 11 11 11 es un número binario de 8 bits, es decir, 1 byte y para obtener en base 10 realizamos la operación $1\times2^7+1\times2^6+1\times2^5+1\times2^4+1\times2^3+1\times2^2+1\times2^1+1\times2^0=256$. Estas tablas hijas (porque su madre es la tabla ASCII) tienen nombres muy específicos definidos por organismos reconocidos como la ISO, IBM, Microsoft, etc. Las más conocidas en Latinoamérica: Latin-1 (ISO/IEC 8858-1), Windows-1252 e IBM 850, esta última porque Windows 10 (versión 1903) para México la tiene como default cuya evidencia al presionar las teclas alt+213 le imprimirá ı (U+0131), de no ser así revísese las tablas de este libro para ver con cuál de ellas coincide su sistema operativo.

Un carácter, al igual que una persona, pertenece a una *familia*, algunos la llaman bloques, tablas, plantillas o Charts (para el idioma ingles).

```
char x = '\u221E';
System.out.printf("block: %s\n", Character.UnicodeBlock.of(x));

/* Lo mismo con la librería de mdt496 (bo.mdt496.unicode)
 * pero esta nos muestra toda la información de un bloque.
 * Además, permite consultar la URL donde reside el PDF del bloque.
 */
System.out.printf("block(mdt496-info): %s\n", MdtUnicodeBlock.of(x).info());
System.out.printf("block(mdt496-url): %s\n", MdtUnicodeBlock.of(x).pdfURL());
```

```
block: MATHEMATICAL_OPERATORS
block(mdt496-info): start=0x2200, range=2200..22FF, name=Mathematical Operators
block(mdt496-url): http://www.unicode.org/charts/U2200.pdf
```

La *categoría* es, al propósito que responde el carácter. Por ejemplo, hay algunos que son de control (Cc) otras son letras (Lo, Ll, Lu), números (No, Nd, Nl), símbolos matemáticos (Sm), etc.

```
char x = '\u221E';

/* El método getType, aunque útil, lastimosamente solo regresa el id */
System.out.printf("Type: %s\n", Character.getType(x));

/* Pero la librería de mdt496 (bo.mdt496.unicode)
```

```
 * si nos regresa el nombre, e incluso toda la información.
 */
System.out.printf("Type(mdt496): %s\n", MdtUnicodeType.of(x));
System.out.printf("Type(mdt496-info): %s\n", MdtUnicodeType.of(x).info());
```

```
Type: 25
Type(mdt496): Math Symbol
Type(mdt496-info): id=25, name=Math Symbol, nameShort=Sm
```

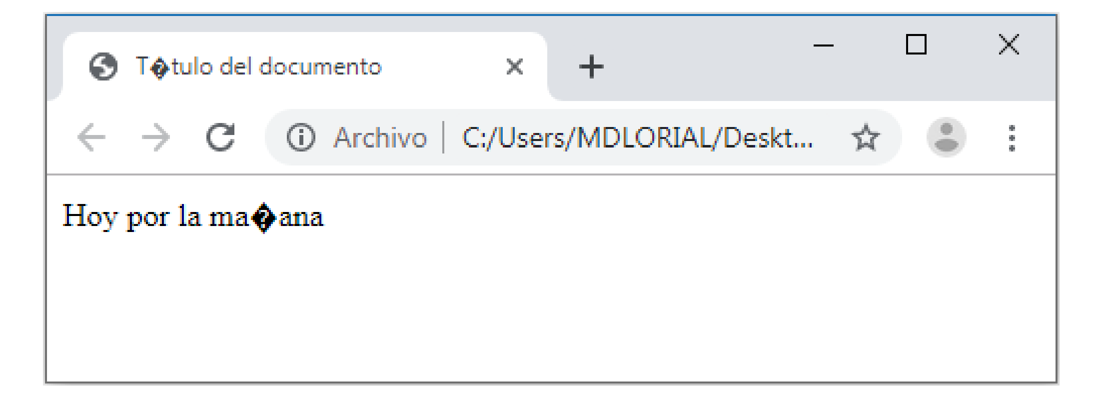
1.2 Importancia

Hay contextos en los cuales es crítico saber cómo funciona la codificación de caracteres. Por ejemplo, si está empleando paquetes ofimáticos tipo WYSIWYG como Microsoft Word u OpenOffice, es el software quien lidia con la codificación de caracteres. En cambio en un entorno de lenguaje de marcado como LaTeX, HTML y el mundo de la programación es el autor quien tiene que proporcionar la información respecto a la codificación de caracteres. Por ejemplo, vea el siguiente código HTML y explique porque la salida (Figura 1.1) imprime caracteres extraños.

Ejemplo 1.1. Declarando el `charset`

```
<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="utf-8">
        <title>Título del documento</title>
    </head>
    <body>
        Hoy por la mañana
    </body>
</html>
```

Figura 1.1. Efectos de una declaración `charset` que no corresponde



Según el software editor (Figura 1.2) el archivo esta codificado en ANSI (azul), pero el programador está declarando en los metadatos que es utf-8 (rectángulo-rojo).

Referencias

[1] ISO/IEC 10646-1, *Information technology–Universal Multiple-Octet Coded Character Set (UCS)–Part 1: Architecture and Basic Multilingual Plane*, 2.ª ed., ISO Geneva, 2000.

[2] ISO/IEC 10646-2, *Information technology–Universal Multiple-Octet Coded Character Set (UCS)–Part 2: Supplementary Planes*, ISO Geneva, 2001.

[3] Unicode Consortium, *The Unicode Standard Version 12.0*, 2019, ISBN: 978-1-936213-22-1. dirección: https://www.unicode.org/versions/Unicode12.0.0/UnicodeStandard-12.0.pdf (visitado 15-06-2019).

[4] J. Gosling, B. Joy, G. Steele, G. Bracha, A. Buckley, D. Smith y G. Bierman, *The Java™ Language Specification, Java SE 15 Edition*, Oracle America, Inc., 10 de ago. de 2020. dirección: https://docs.oracle.com/javase/specs/jls/se15/jls15.pdf (visitado 05-10-2020).

[5] A. Jeffrey y F. Mittelbach, *inputenc.sty*, v1.2c, 2015.

[6] T. Oetiker, H. Partl, I. Hyna y E. Schlegl, *Introducción no tan corta a LaTeX*. 2014, Versión: 5.03.

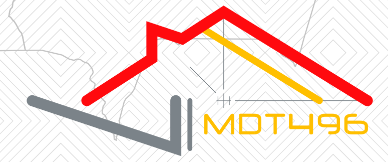
[7] V. Volovich, W. Lemberg y LaTeX3 Project Team, *Cyrillic languages support in LaTeX*, 1999. dirección: http://linorg.usp.br/CTAN/macros/latex/base/cyrguide.pdf (visitado 09-12-2020).



Contacto

MDT496 ESTUDIOS | Marco Mendieta Parihuancollo

www.mdt496.wix.com/live
mdt496@gmail.com
www.youtube.com/mdt496



MDT496 ESTUDIOS

EBD SYSTEM

La Paz - Bolivia